Contents

Kubernetes Ingress

3 minute read ✔ page test

This task describes how to configure Istio to expose a service outside of the service mesh cluster, using the Kubernetes Ingress Resource.

> Using the Istio Gateway, rather than Ingress, is recommended to make use of the full feature

set that Istio offers, such as rich traffic management and security features.

Before you begin

Follow the instructions in the Before you begin and Determining the ingress IP and ports sections of the Ingress Gateways task.

Configuring ingress using an Ingress resource

A Kubernetes Ingress Resources exposes HTTP and HTTPS routes from outside the cluster to services within the cluster.

Let’s see how you can configure a `Ingress` on port 80 for HTTP traffic.

1. Create an `Ingress` resource:

```
$ kubectl apply -f - <<EOF
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  annotations:
    kubernetes.io/ingress.class: istio
  name: ingress
spec:
  rules:
  - host: httpbin.example.com
    http:
      paths:
      - path: /status/*
        backend:
          serviceName: httpbin
          servicePort: 8000
EOF
```

The `kubernetes.io/ingress.class` annotation is

required to tell the Istio gateway controller that it should handle this `Ingress`, otherwise it will be ignored.

2. Access the *httpbin* service using *curl*:

```
$ curl -s -I -HHost:httpbin.example.com "http://$INGRESS_HO
ST:$INGRESS_PORT/status/200"
HTTP/1.1 200 OK
server: istio-envoy
...
```

Note that you use the `-H` flag to set the *Host* HTTP header to “httpbin.example.com”. This is needed because the `Ingress` is configured to handle “httpbin.example.com”, but in your test

environment you have no DNS binding for that host and are simply sending your request to the ingress IP.

3. Access any other URL that has not been explicitly exposed. You should see an HTTP 404 error:

```
$ curl -s -I -HHost:httpbin.example.com "http://$INGRESS_HO
ST:$INGRESS_PORT/headers"
HTTP/1.1 404 Not Found
...
```

Next Steps

TLS

`Ingress` supports `specifying TLS settings`. This is supported by Istio, but the referenced `Secret` must exist in the namespace of the `istio-ingressgateway` deployment (typically `istio-system`). `cert-manager` can be used to generate these certificates.

Specifying path type

By default, Istio will treat paths as exact matches,

unless they end in `/*` or `.*`, in which case they will become prefix matches. Other regular expressions are not supported.

In Kubernetes 1.18, a new field, `pathType`, was added. This allows explicitly declaring a path as `Exact` or `Prefix`.

Specifying `IngressClass`

In Kubernetes 1.18, a new resource, `IngressClass`, was added, replacing the `kubernetes.io/ingress.class`

annotation on the `Ingress` resource. If you are using this resource, you will need to set the `controller` field to `istio.io/ingress-controller`. For example:

```
apiVersion: networking.k8s.io/v1beta1
kind: IngressClass
metadata:
  name: istio
spec:
  controller: istio.io/ingress-controller
---
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: ingress
spec:
  ingressClassName: istio
  rules:
```

```
  - host: httpbin.example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          serviceName: httpbin
          servicePort: 8000
```

Cleanup

Delete the `Ingress` configuration, and shutdown the `httpbin` service:

```
$ kubectl delete ingress ingress
$ kubectl delete --ignore-not-found=true -f @samples/httpbin/htt
pbin.yaml@
```